Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 3s (2025)

Intelligent Railways: Leveraging Retrieval-Augmented Generation for Smarter Systems

Gresha Bhatia[1], Rohini Temkar[2], Aryan Raje[3], Arya Raje[4] , Ishita Marathe[5], Prasad Lahane[6]

gresha.bhatia@ves.ac.in[1] , rohini.temkar@ves.ac.in[2], d2021.aryan.raje@ves.ac.in[3], d2021.arya.raje@ves.ac.in[4], d2021.ishita.marathe@ves.ac.in[5], d2021.prasad.lahane@ves.ac.in[6]

[1,2,3,4,5] Department of Computer Engineering , VESIT, Mumbai, Maharashtra, India

Article History:

Received: *20-09-2024*

Revised: *03-11-2024*

Accepted: *17-11-2024*

Abstract:

In an era for faster, secure and convenient mode of travel, there is a need for a system that provides real time updates This paper presents a technical study on the integration of Retrieval-Augmented Generation (RAG) systems within railway operations, emphasizing their potential to enhance decision-making, service delivery, and passenger engagement. The study explores how RAG systems can streamline processes by providing accurate, context-aware responses to inquiries across various railway services, including ticketing, scheduling, and customer support. The findings highlight key challenges such as data security, infrastructure limitations, and the necessity for specialized training, while also emphasizing the operational benefits, including improved efficiency and greater accessibility of services. The methodology encompasses a comprehensive review of existing RAG implementations in transportation, followed by the design and analysis of a prototype system specifically tailored to railway needs, utilizing domain-specific datasets and natural language queries. This study offers valuable insights into the feasibility and scalability of RAG systems for enhancing the efficiency and responsiveness of railway operations.

Keywords: Retrieval Augmented Generation, Railway system, Public Sector, large language model, artificial intelligence , machine learning and natural language processing algorithms.

I.INTRODUCTION

As per the India Brand Equity Foundation (IBEF) site, the rate at which artificial intelligence has been adopted is approximately 48 % in the FY 2024 and is expected to grow by another 5 -7 % in the upcoming year. One of the major focuses for development is on the Infrastructure need. Figure 1 below represents the passenger volume over years. As population grows as well as economic development, there is a need to have improved transport infrastructure. Utilization of state of the art technology in the public transport systems such as the roads, railways, aviation and shipping forms the need of the hour. This paper elaborates on the use of Retrieval Augmented Generation (RAG) for the railway sector as a transformative application in travel and transportation. The rapid evolution of Generative AI (GenAI) and highly intelligent large language models (LLMs) has revolutionized information retrieval processes, enabling more precise and contextualized responses. These advancements have given rise to RAG as an emerging next-generation solution, blending real-time retrieval of domain-specific information with the generative capabilities of LLMs.

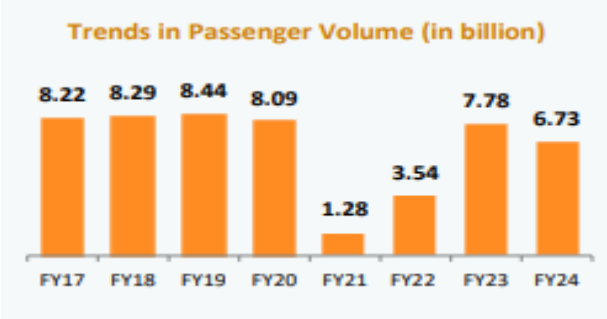

Figure 1: Passenger volume over years.

Source:- https://www.ibef.org/industry/indian-railways/infographic

The paper further elaborates on the related work done in this domain in section II. The limitations observed in the existing systems are expressed in section III. This is then followed by the process of retrieval augmentation generation in section IV. The queries posted and the answers generated through the RAG component forms a crucial part for the users as the answers generated give a real time scenario of the arrival and departure of the trains on a specific platform . This is elaborated in section V through the algorithm and the process design section . Implementation of the proposed system through various natural language processing and deep learning algorithms such as Open AI, Google Generative AI, Hugging face are focussed upon in section VI along with machine learning algorithms. Evaluation measures are elaborated upon in section VII . This is followed by7 the results of the proposed system . Section VIII and Section IX then represent the conclusion and the future work respectively .

II.RELATED WORK

The literature survey undertaken for the above mentioned problem consists of understanding that railways being a part of the public sector, has to have a lot of real time , accurate , precise and day to day information to be stored, processed and analyzed. These vast datasets are crucial for decision-making and service delivery, Thus it was observed that RAG systems combine retrieval-based and generative models to provide accurate, contextually aware responses. With platforms like OpenAI reaching over 100 million active users within months of launch, the growing reliance on AI-driven tools was evident. This study therefore addresses the inefficiencies in traditional public sector information retrieval, which often leads to delays and lack of personalization. In paper [4] the authors highlighted the logistical and financial decline of Indian Railways, emphasizing problems like a poor Operating Ratio (OR), Cost of Revenue (COR), and Return on Logistics Assets Ratio (ROLAR), signaling a need for deeper analysis of these issues. In [7] the paper identifies common failure points when engineering RAG systems in diverse domains (research, education, and biomedical), stressing the challenges and necessary refinements for robust system performance. In [1] the research introduces the RAG model, utilizing pre-trained seq2seq models (parametric memory) and dense vector indexing of Wikipedia (non-parametric memory) for knowledge-intensive NLP tasks. Paper [8] focussed on fine-tuning RAG architectures, particularly the DPR retriever, while addressing the engineering complexities that arise in end-to-end tuning for effective question-answering. In paper [9] the authors demonstrated the superior performance of RAG-based agents over existing LLMs in science-based QA benchmarks, underscoring its efficacy in scientific research applications. Paper [10] offered a

comprehensive review of the advancements in RAG systems, exploring critical components, evaluation frameworks, and cutting-edge technologies.FlashRAG toolkit that supported modular and efficient RAG research through 12 advanced RAG methods was elaborated on in paper [11] . This had a rich collection of datasets, and comprehensive evaluation metrics.

III.LIMITATIONS IN EXISTING SYSTEM

By exploring the potential of RAG to enhance accuracy, efficiency, and user experience across domains such as healthcare, education, and government services, the paper aims to propose a framework for integrating RAG systems into public sector operations to improve service delivery and decision-making. The limitations observed in the existing systems include :-

1. Context Management for Long Documents

Existing RAG systems may struggle with efficiently managing long contexts, which is crucial for handling detailed documentation, reports and real-time updates across various public services.

2. Robustness to Misinformation

If a RAG system retrieves outdated or incorrect information, it could lead to inaccurate responses for citizens seeking assistance with services such as healthcare, social welfare, or public safety.

3. Integration of Domain-Specific Knowledge

Current RAG systems may not be well-adapted to the specific needs of different public sector agencies, such as handling diverse and localized information or integrating with various live data sources relevant to specific public services.

4. Resource-Intensive Operations

High computational costs and memory requirements could hinder the feasibility of deploying RAG systems at scale for real-time updates and queries in public sector operations, especially in resource-constrained environments.

5. Generalization and Domain Adaptation

Existing RAG methods do not perform optimally across different public sector domains, affecting the system's ability to effectively handle the varied queries and requirements of citizens interacting with government services.

This gap underscores the need for further exploration to develop effective frameworks for RAG deployment in public services.

IV.RETRIEVAL AUGMENTED GENERATION

Retrieval-Augmented Generation (RAG) forms an innovative framework that combines the strengths of two fundamental components: a retrieval system and a generation system. This hybrid approach enhances the ability to provide accurate and contextually relevant responses to complex queries, making it particularly valuable in various applications including public sector operations. The details of both the modules are elaborated below and represented in figure 2.

1. Retrieval System

The retrieval component is responsible for sourcing relevant information from a predefined database or knowledge base. It utilizes techniques such as keyword matching, semantic search, and information retrieval algorithms to identify documents or data snippets that relate closely to the user's query. By retrieving pertinent information, this system lays the groundwork for generating informed responses. The effectiveness of the retrieval system is crucial, as it ensures that the generative model operates with accurate and relevant content.

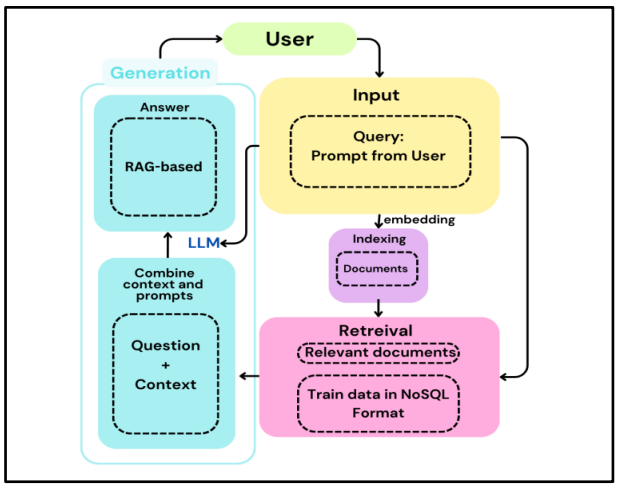

Figure 2 : Block Diagram of RAG

2. Generation System

The generation component leverages advanced Natural Language Processing (NLP) techniques, often powered by large language models (LLMs), to create coherent and contextually appropriate responses. Once the retrieval system has identified relevant information, the generation model synthesizes this data, generating responses that are not only informative but also human-like in their structure and tone. This component enhances the user experience by providing answers that feel natural and engaging, while also incorporating the specific details retrieved from the database.

Together, these components form a robust RAG system that effectively addresses the challenges of information retrieval and response generation, making it a powerful tool for improving interactions in various domains, including healthcare, education, and public sector services. By integrating retrieval and generation capabilities, RAG systems can offer users more personalized and accurate information, enhancing decision-making and overall service delivery.

The elaborated description for the same is as follows.

User Query: The initial input provided by the user.

Generation Answer: A preliminary response generated by the language model (LLM) based on internal knowledge.

Embedding: Converts the user query into an embedding format suitable for retrieval.

Indexing: Organizes and stores documents, typically in NoSQL format, for efficient access.

Retrieval: Searches the indexed documents to identify relevant ones based on the embedded query.

Relevant Documents: The documents identified as containing pertinent information.

Combine Context and Prompts: Integrates the relevant documents with the original query and context.

RAG-based Generation: The LLM processes the enriched query to generate a final, contextually informed response.

Output: The final, accurate answer is delivered to the user.

V. ALGORITHM AND PROCESS DESIGN

Retrieval-Augmented Generation (RAG) system, algorithm and process design are crucial for transforming user queries into meaningful outputs. This involves a series of carefully crafted steps—from converting user input into embeddings, retrieving relevant data from structured sources, to generating natural language responses using advanced language models like GPT-2. Each stage is meticulously designed to ensure accurate data retrieval, efficient processing, and coherent response generation**.**

The integration of a RAG system begins with a JSON file containing station data, which serves as the primary source of structured information about the various stations under consideration. When a user submits a query, the system converts this query into embeddings. This user input is further transformed into a vector representation that facilitates efficient comparison with existing data. The system then extracts relevant text from the JSON file, retrieving pertinent station information that matches the user's query. Utilizing the FAISS (Facebook AI Similarity Search) framework, it then retrieves similar documents. This process searches the vector database to identify entries closely aligned with the user's input. Following this, the system generates embeddings for the extracted station data, creating vector representations that can be compared with the query embeddings, as represented in figure 3.

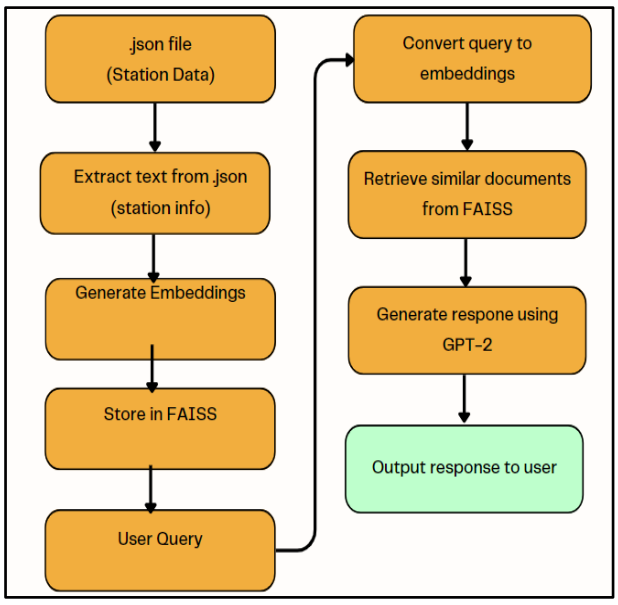

Figure 3 represents the block diagram for RAG systems

The heart of the process lies in the response generation using GPT-2, where the model leverages the retrieved data to formulate a coherent and contextually relevant natural language response. Additionally, to optimize future interactions, the system stores the embeddings of the station data in FAISS, ensuring quick access for subsequent queries. Finally, the system outputs the response to the user, presenting the generated answer based on their original inquiry, thereby enhancing the overall user experience.

VI. IMPLEMENTATION DETAILS

In the rapidly evolving landscape of artificial intelligence (AI), various platforms offer unique capabilities that cater to different user needs and applications. This comparison focuses on three prominent AI platforms: OpenAI, Google Generative AI, and Hugging Face. Each platform presents distinct features, advantages, and limitations that impact their usability and effectiveness in various contexts, particularly in the realm of natural language processing (NLP) and machine learning.

1. OpenAI

OpenAI is renowned for its sophisticated language models, particularly the GPT series, which are adept at understanding and generating human-like text. These models can be applied across various domains, including chatbots, content generation, and more. OpenAI offers a comprehensive API that enables developers to seamlessly incorporate these models into their applications.

Disadvantages:

Cost: Utilizing OpenAI's API can be expensive, especially for projects that demand high interaction levels or large volumes of queries.

Usage Limits: The free tier comes with stringent usage restrictions, which may limit experimentation and development for students or smaller initiatives.

Complexity: While the models are powerful, the intricacies involved in their usage may pose challenges for those who are new to the technology..

2. Google Generative AI

Google provides a suite of Generative AI models tailored for tasks like content creation and code generation. These models benefit from Google’s vast data and advanced machine learning infrastructure, yielding high-quality results across a variety of applications.

Disadvantages:

Cost: Similar to OpenAI, using Google’s models can lead to considerable expenses, particularly in business environments.

Usage Limits: Google also places restrictions on free usage, which could limit the capacity to conduct extensive testing.

Integration Challenges: Implementing and integrating Google’s AI solutions can be intricate, often requiring a deeper understanding of their ecosystem.

3. Hugging Face

Hugging Face is an open-source platform that offers a wide range of pre-trained models for tasks in natural language processing and beyond. The Hugging Face Transformers library allows users to easily access and customize these models, making it appealing to both novices and seasoned users.

Reasons for Choosing Hugging Face:

Cost-Effective: Models on Hugging Face are free to use, making them accessible for students and those with limited budgets.

Open Source: The platform fosters collaboration and resource sharing, enabling students to engage with a lively community and share their discoveries.

Flexibility: Users can fine-tune models for specific applications without incurring significant costs, promoting extensive experimentation and learning.

Table 1 below gives a comparison of various language models that have been identified for the experimental set up . It further elaborates on the reason behind selecting the Hugging model for implementation purpose.

Characteristics	**OpenAI**	**GoogleAI**	**Hugging Face**
Pricing	Paid	Paid	Free
Quality	High	Medium	Low
Conversational Capability	Best	Moderate	Low
Token Limit	4000	24000	Manual setting
Quota Impact	Working solution not implemented due to quota limit	Working solution not implemented due to quota limit	Working preliminary solution is implemented using 'gpt2' model

Table 1. Comparison of language models

Algorithms Used

1. Recommendation Algorithms

a. Collaborative Filtering:

i.User-Based Collaborative Filtering:

```
function user_based_collaborative_filtering(user_id):
    similar_users = find_similar_users(user_id)
    recommended_items = []
    for user in similar_users:
    for item in user.recommended_items:
    if item not in user.past_items:
```

recommended_items.append(item)

return unique(recommended_items)

ii.Item-Based Collaborative Filtering:

function item_based_collaborative_filtering(item_id):

similar_items = find_similar_items(item_id)

return user_ratings_for(similar_items)

b. Content-Based Filtering:

i.TF-IDF (Term Frequency-Inverse Document Frequency):

function calculate_tfidf(document, corpus):

tf = term_frequency(document)

idf = inverse_document_frequency(corpus)

tfidf = tf * id return tfidf

ii.Cosine Similarity:

function cosine_similarity(vector_a, vector_b): dot_product = sum(a * b for a, b in zip(vector_a, vector_b))

magnitude_a = sqrt(sum(a * a for a in vector_a))

magnitude_b = sqrt(sum(b * b for b in vector_b))

if magnitude_a == 0 or magnitude_b == 0:

return 0

return dot_product / (magnitude_a * magnitude_b)

Figure 3.1. further elaborates on the working of the system through the flowchart representation

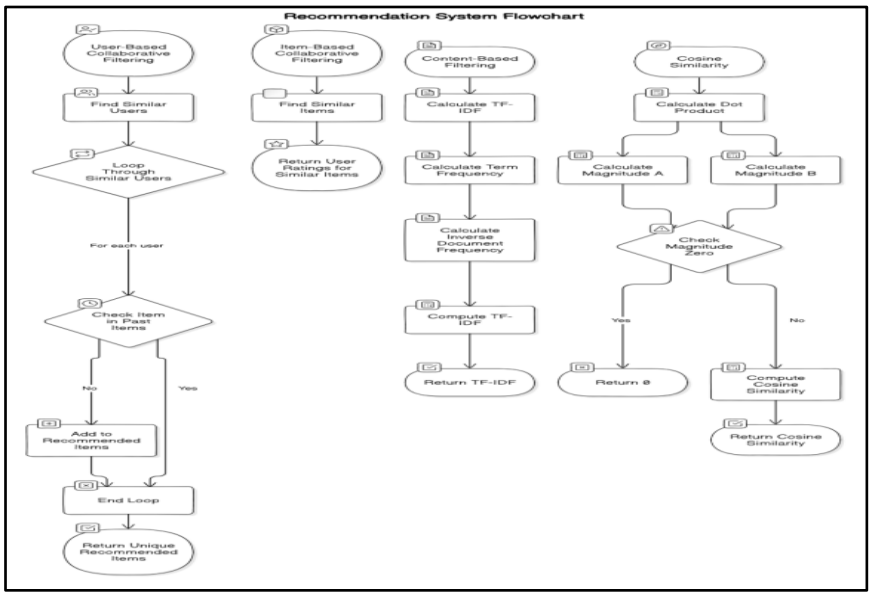

Fig 3.1 Recommendation System Flowchart

Machine Learning Algorithms

Purpose of the machine learning algorithms used was to predict train delays and user behavior based on historical data. The various algorithms considered included:

1.Regression Analysis:

These algorithms are used to determine the trend or pattern for any given situation . For the railways problem too, the regression algorithms considered are the linear and logistic regressions as elaborated below.

a. Linear Regression:

As represented in fig 4.1, the linear regression is implemented through the function mentioned next.

```
function linear_regression(training_data):
    coefficients = initialize_coefficients()
    for epoch in range(num_epochs):
        for data_point in training_data:
            prediction = predict(data_point.features, coefficients)
            error = prediction - data_point.target
            update_coefficients(coefficients, error, data_point.features)
    return coefficients
```

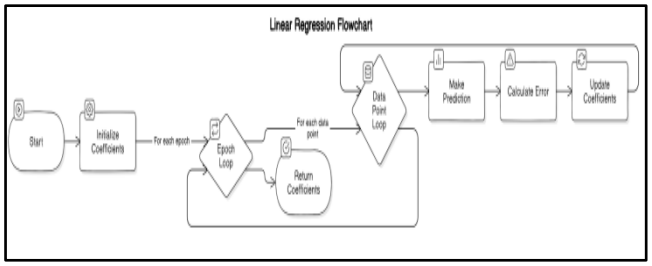

Fig 4.1 Linear Regression Algorithm

b. Logistic Regression:

Using the logarithmic function , the logistic regression technique is incorporated as indicated below in figure 4.2

```
function logistic_regression(training_data):
    coefficients = initialize_coefficients()
    for epoch in range(num_epochs):
        for data_point in training_data:
```

```
        prediction = sigmoid(predict(data_point.features, coefficients))
        error = prediction - data_point.target
        update_coefficients(coefficients, error, data_point.features)
    return coefficients
```

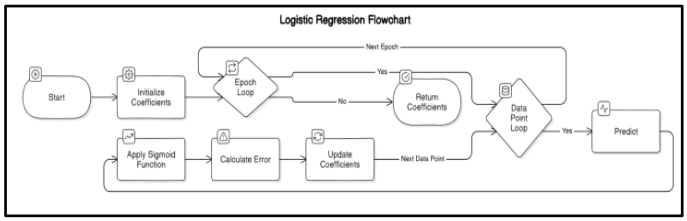

Fig 4.2 Logistic Regression Algorithm

c. Decision Trees:

CART (Classification and Regression Trees):

This is represented through the fig 4.3 and its implementation is observed through the algorithm steps mentioned below

```
function cart(training_data):
    if stopping_condition_met(training_data):
        return create_leaf_node(training_data)
    best_split = find_best_split(training_data)
    left_data, right_data = split_data(training_data, best_split)
    node = create_internal_node(best_split)
    node.left = cart(left_data)
    node.right = cart(right_data)
    return node
```

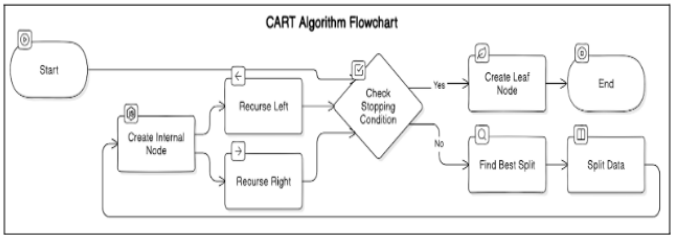

Fig 4.3 CART Algorithm

d.Natural Language Processing (NLP) Algorithms

Fig 4.4 represents the tokenization and NER used in the proposed RAG systems.

i. Tokenization:

```
function tokenize(text):
```

```
    return text.split() // split text by whitespace
```

ii. Named Entity Recognition (NER):

```
function named_entity_recognition(text):
    entities = []
    for word in tokenize(text):
        if is_entity(word):
     entities.append(word)
    return entities
```

iii. Sentiment Analysis:

```
function sentiment_analysis(text):
    score = 0
    for word in tokenize(text):
        score += sentiment_score(word)
    if score > 0:
        return "Positive"
    else if score < 0:
        return "Negative"
    else: return "Neutral"
```

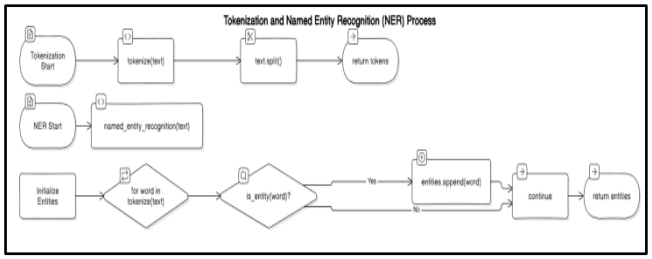

Fig 4.4 Tokenization and NER

VII. RESULTS AND DISCUSSION

When querying railway stations within the Konkan railway (KR) zone, the system successfully identified several key stations, including Kumta (KT), Veer (VEER), and Kudal (KUDL). Other notable stations within the same zone include Khed (KHED), Kalambani (KLMB), and Indapur (INP). Additionally, stations such as Kamthe (KMAH), Kharepatan (KRPM), and Kankavli (KKW) were also recognized as part of the KR zone.

In a separate inquiry regarding the station 'Roha' (ROHA), the system provided clear and concise information indicating that it is part of the Central Railway (CR) zone. These results highlight the effectiveness of the system in retrieving accurate and relevant data based on user queries, showcasing

its potential to assist users in navigating railway services efficiently. The integration of such features not only enhances user experience but also promotes accessibility to vital transportation information across different railway zones.

Figures 5.1 to 5.5 specify the detailed results obtained .

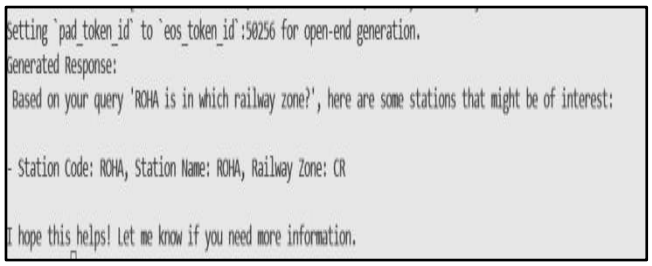

Figure 5.1

Figure 5.2

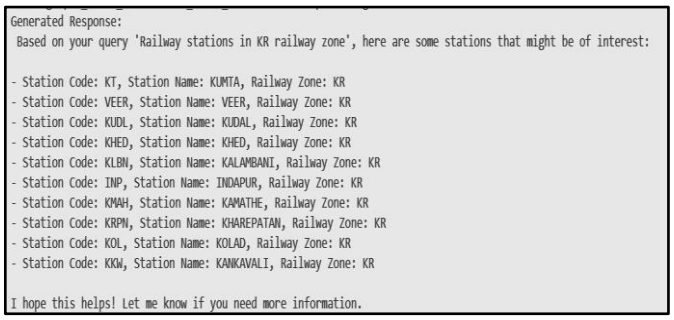

Figure 5.3

Figure 5.4

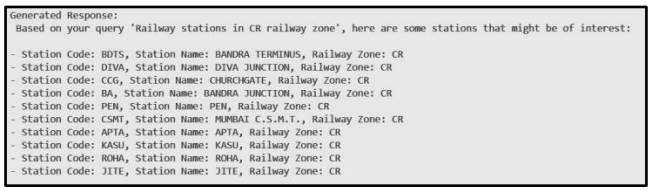

Figure 5.5

VIII. CONCLUSION

The development and implementation of a Retrieval Augmented Generation (RAG) system for the railway sector represents a significant advancement in how data is managed, analyzed, and reported. By combining sophisticated retrieval techniques with powerful generative language models, this paper addresses critical challenges faced by the railway industry in handling large volumes of diverse data.By integrating advanced retrieval and generative capabilities, the RAG system not only enhances the efficiency of data processing but also supports a more proactive and responsive approach to managing railway operations. This innovative solution is poised to transform how the railway sector leverages data, ultimately contributing to more effective and efficient railway operations and improved service quality.

REFERENCES

[1] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, Douwe Kiela, “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks”, 22 May 2020 (v1), 12 Apr 2021 (this version, v4)

[2] Isamu Isozaki, “Literature Review on RAG(Retrieval Augmented Generation) for Custom Domains”, Nov 26, 202 [3] Kieran Pichai, “A Retrieval-Augmented Generation Based Large Language Model Benchmarked On a Novel Dataset”, November 2023, Journal of Student Research, DOI:10.47611/jsrhs.v12i4.6213, License :CC BY-NC-SA 4.0

[4] Shouvik Sanyal, Alam Ahmad, Hafiz Wasim Akram, “An Analysis of Performance of Indian Railways”, January 2021, DOI:10.1504/IJLSM.2021.10043738

[5] Mohd Arshad, Muqeem Ahmed, “Prediction of Train Delay in Indian Railways through Machine Learning Techniques ”, February 2019, International Journal of Computer Sciences and Engineering 7(2):405-4117(2):405-411, DOI:10.26438/ijcse/v7i2.405411.

[6] Mohd Arshad, Muqeem Ahmed, “Train Delay Estimation in Indian Railways by Including Weather Factors Through Machine Learning Techniques”, September 2019, DOI:10.2174/2666255813666190912095739.

[7] Scott Barnett, Stefanus Kurniawan, Srikanth Thudumu, Zach Brannelly, Mohamed Abdelrazek, “Seven Failure Points When Engineering a Retrieval Augmented Generation System”, 2024 IEEE/ACM 3rd International Conference on AI Engineering – Software Engineering for AI (CAIN), 18 June, 2024

[8] Shamane Siriwardhana, Rivindu Weerasekera, Elliott Wen, Suranga Nanayakkara, “FINE-TUNE THE ENTIRE RAG ARCHITECTURE (INCLUDING DPR RETRIEVER) FOR QUESTION-ANSWERING”, arXiv:2106.11517v1, 23 June, 2021

[9] Jakub Lala Odhran O’Donoghue, Aleksandar Shtedritski, Sam Cox, Samuel G Rodriques, Andrew D White, “PaperQA: Retrieval-Augmented Generative Agent for Scientific Research”, arXiv:2312.07559v2 , 14 December, 2023

[10] Yunfan Gaoa , Yun Xiongb , Xinyu Gaob , Kangxiang Jiab , Jinliu Panb , Yuxi Bic , Yi Daia , Jiawei Suna , Meng Wangc , and Haofen Wang, “Retrieval-Augmented Generation for Large Language Models: A Survey”, arXiv:2312.10997v5 [cs.CL] , 27 March, 2024

[11] Jiajie Jin, Yutao Zhu, Xinyu Yang, Chenghao Zhang, Zhicheng Dou,”FlashRAG: Modular Toolkit for Efficient Retrieval-Augmented Generation Research”, arXiv:2405.13576v1 [cs.CL], 22 May, 2024

[12] Chidaksh Ravuru, Sagar Srinivas Sakhinana, Venkataramana Runkana, “Agentic Retrieval-Augmented Generation for Time Series Analysis”, arXiv:2408.144846v1 [cs.CL], 18 Aug, 2024

[13] Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, Zhicheng Dou, “MemoRAG: Moving Towards Next-Gen RAG via Memory-Inspired Knowledge Discovery”, arXiv:2409.055916v2 [cs.CL], 10 Sept, 2024

[14] Zahra Sepasdar, Sushant Gautam, Cise Midoglu, Michael A. Riegler, Pål Halvorsen, “Enhancing Structured-Data Retrieval with GraphRAG: Soccer Data Case Study”, arXiv:2409.17580v1 [cs.IR], 26 Sept, 2024

[15] Jordi Bayarri-Planas, Ashwin Kumar Gururajan, Dario Garcia-Gasulla, “Boosting Healthcare LLMs Through Retrieved Context”, arXiv:2409.13127v1 [cs.AI], 23 Sept, 2024

[16] Satyapriya Krishna, Kalpesh Krishna, Anhad Mohananey, Steven Schwarcz, Adam Stambler, Shyam Upadhyay, Manaal Faruqui, “Fact, Fetch, and Reason: A Unified Evaluation of Retrieval-Augmented Generation”, arXiv:2409.12941 [cs.CL], 19 Sept, 2024

[17] Yue Yu, Wei Ping, Zihan Liu, Boxin Wang, Jiaxuan You, Chao Zhang, Mohammad Shoeybi, Bryan Catanzaro, “RankRAG: Unifying Context Ranking with Retrieval-Augmented Generation in LLMs”, arXiv:2407.02485v1 [cs.CL], 19 Sept, 2024